

PREDICTING CO₂ EMISSION IN VEHICLES USING MACHINE LEARNING

A UG PROJECT PHASE-1 REPORT

Submitted to

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD**

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

AMARAVADI JAVALI	19UK1A05J1
LINGALA SHALINI	19UK1A05J8
KATABATHINI BHARATH	19UK1A05L2
JAMPALA SAITEJA	19UK1A05J3

Under the esteemed guidance of

Mr. T. SANATH KUMAR

(Assistant Professor)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE**

(Affiliated to JNTUH, Hyderabad)

Bollikunta, Warangal-506005

2019– 2023

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE
BOLLIKUNTA, WARANGAL – 506005
2019 – 2023**



CERTIFICATE OF COMPLETION OF
A UG PROJECT PHASE-1

This is to certify that the Major project entitled “**PREDICTING CO2 EMISSION IN VEHICLES USING MACHINE LEARNING**” is being submitted by **A.JAVALI (19UK1A05J1),L.SHALINI (19UK1A05J8),K.BHARATH (19UK1A05L2),J.SAITEJA (19UK1A05J3)** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering to Jawaharlal Nehru Technological University Hyderabad** during the academic year **2022-23**, is a record of workcarried out by them under the guidance and supervision.

Project Guide

Mr. T. SANATH KUMAR
(Assistant Professor)

Head of the Department

Dr. R. NAVEEN KUMAR
(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. PRASAD RAO**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-I in the institute.

We extend our heartfelt thanks to **Dr. R. NAVEEN KUMAR**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-I.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-I and for their support in completing the UG Project Phase-I.

We express heartfelt thanks to the guide, **Mr. T. SANATH KUMAR** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-I.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

A.JAVALI **(19UK1A05J1)**

L.SHALINI **(19UK1A05J8)**

K.BHARATH **(19UK1A05L2)**

J.SAITEJA **(19UK1A05J3)**

ABSTRACT

Our personal vehicles are a major cause of global warming. Collectively, cars account for nearly one-fifth of all emissions, emitting around 24 pounds of carbon dioxide and other global-warming gases for every gallon of gas. About five pounds comes from the extraction, production, and delivery of the fuel, while the great bulk of heat-trapping emissions—more than 19 pounds per gallon—comes right out of a car's tailpipe.

It is assumed the average gasoline vehicle on the road today has a fuel economy of about 22.0 miles per gallon and drives around 11,500 miles per year. Every gallon of gasoline burned creates about 8,887 grams of CO₂. Twenty years ago (1998) the car industry agreed to a voluntary commitment to reduce new car emissions by 25% by 2008.¹⁰ Then, CO₂ emissions on the road from new cars were around 203g/km.¹¹ Today, they are still around 170g/km and unlikely to reach 140g/km until after 2020.

Adequate CO₂ is essential for vegetation, but industrial chimneys and land, space and oceanic vehicles exert tons of excessive CO₂ and are mostly responsible for the greenhouse effect, global warming and climate change. Due to COVID-19, CO₂ emission was in 2020 at its lowest level compared to prior decades. However, it is unknown how long it will take to reduce CO₂ emission to a tolerable point. Furthermore, it is also unknown to what extent it can increase or change in the future.

This study successfully predicts CO₂ emission either for the COVID-19 period or the post-COVID-19 normal periods. The Machine Learning (ML) method used in this study has shown good agreement with the IPCC model in predicting the past emissions, the current emissions due to COVID-19 and the emissions of the upcoming future. These prediction results can be an asset for the decision support system to develop a suitable policy for global CO₂ emission reduction.

For future research, a number of other external influence variables responsible for CO₂ emission can be added for finer forecasts. This research is an original work in predicting COVID-19-affected CO₂ emission using AI through the ML methodology.

Keywords: Artificial Intelligence; machine learning; CO₂ emission; global warming; atmosphere monitoring; atmosphere maintenance.

TABLE OF CONTENTS:

1. INTRODUCTION.....	1
OVERVIEW	1
PURPOSE.....	1
2. LITERATURE SURVEY	2-5
EXISTING PROBLEM	2-3
PROPOSED SYSTEM	3-5
3. THEORITICAL ANALYSIS	6-8
BLOCK DIAGRAM.....	6
SOFTWARE DESIGNING.....	7-8
4. EXPERIMENTAL INVESTIGATIONS.....	9
5. CONCLUSION	10
6. FUTURE SCOPE.....	11

1. INTRODUCTION

1.1 OVERVIEW:

With the development of the world economy, the situation of carbon dioxide emissions controls becoming increasingly serious. CO₂ emissions are the primary driver of global climate change. It is widely recognized to avoid the worst impacts of climate change. A typical vehicle emits about 4.6 metric tons of carbon dioxide per year.

This causes global warming and effects the environment. The number can vary based on a vehicle's fuel, fuel economy, and the number of miles driven per year. When gasoline in the vehicle burns, the carbon and hydrogen separate.

The hydrogen combines with oxygen to form water (H₂O), and carbon combines with oxygen to form carbon dioxide (CO₂). This is how CO₂ is emitted from vehicles.

This project will be predicting the amount of CO₂ released by the vehicles by exploring the inputs of the vehicle such as engine-size, fuel-consumption etc.

1.2 PURPOSE:

The goal of the project is to develop an emission predictor which predicts the amount of the CO₂ emitted by a vehicle with certain characteristics. The model will be trained and tested using Machine Learning Techniques.

The predictor is built by using Flask applications. HTML and CSS are used for creating and designing the webpage.

2. LITERATURE SURVEY

EXISTING PROBLEM:

CO₂ emissions are highly renewable, qualitative and quantitative. Exposure to high CO₂ levels in vehicles results in unpleasant feeling, fatigue, drowsiness or lethargy among the drivers and passengers. CO₂ emitted from vehicles can be reduced little by air filtration.

The CO₂ which is emitted from vehicles causes climate change by trapping heat, and also leads to respiratory diseases in the human beings. Extreme weather, food supply disruptions, and increased wildfires are few effects of climate change caused by CO₂ emissions.

Carbon emissions include a number of different chemicals and particulates that are produced when fuel is burned in an engine. Some of the major substances found in a car's exhaust include carbon dioxide, ozone, and carbon monoxide. Other chemicals often found in exhaust gasses include benzene and nitrogen oxides.

Many of these chemicals serve an important purpose in different parts of the atmosphere, but they can have bad consequences when human beings inhale them directly. Mitigation of Carbon Dioxide emission is the challenge of the future in order to stabilize global warming. The factors affecting CO₂ emissions of vehicles include the GDP, population, urbanization rate, transportation development level, transportation energy intensity, energy consumption structure, and industrial structure.

The level of traffic development is a comprehensive indicator, and there exist some differences in how to measure it quantitatively. The classical regression analysis requires the independent variables to be linearly independent.

The carbon emissions are calculated according to the total amount of smoke. The amount of CO₂ emission from the transport sector (including cars) accounts for about 20% of total CO₂ emissions. Accordingly, from the viewpoint of preventing global warming, reducing that proportion is a key issue.

In regard to CO₂ emissions from cars, fuel economy standards are getting tougher all over the world, so improving the fuel economy of cars is strongly desired. It is considered that the fuel economy of engines will be further improved by boosting engine efficiency and by hybridization or electrification of cars.

Improving fuel economy by improving “driving operation” (i.e. the operation in which a car is driven) and by smoothing traffic flows will come into the picture in the near future.

Vehicle emissions have a number of harmful effects on human health. Exposure to car emissions increases the risk of getting certain cancers. Carbon traps heat in the atmosphere, preventing it from escaping from the earth. This has led to a warmer earth over the past century, increasing the odds of severe weather patterns, droughts, and other problems.

This problem can be rectified by using Machine Learning Techniques by predicting the amount of CO₂ emitted from a vehicle by giving few inputs or features of the vehicle. By doing this prediction, the amount of CO₂ from the vehicles can be known and reduced by doing few measures.

PROPOSED SOLUTION:

Significantly reducing CO₂ emissions from cars will not be easy, but the available data can be used to extract the features, know the behavior of cars, and try to reduce the emissions. Machine Learning techniques can be used in this regard. The solution to the above discussed problems is to build a predictor which predicts the amount of CO₂ released by a vehicle. The vehicle CO₂ emission model is derived based on the theory of vehicle dynamics. When the CO₂ emission predictor predicts the amount of CO₂ released, then the car might be modified or replaced with parts which cause less amount of CO₂ to be released.

In this project, we take a dataset which contains features of the vehicles like:

- Type of car
- Car class
- Engine size
- Cylinder size
- Transmission of the car
- Fuel Type
- Fuel Consumption
- Combine Fuel Consumption
- Fuel Consumption highway

Now using machine learning techniques like Linear Regression, we prepare the model. The following steps are to be performed to the dataset:

- Download the dataset: Machine Learning depends heavily on data, without data, it is impossible for a machine to learn. It is the most crucial aspect that makes algorithm training possible.
- Pre-process the data:
 - contains handling the null values
 - handling the categorical values if the dataset contains
 - normalizing the data wherever required
 - identifying the independent and dependent variables in the dataset
 - Splitting the dataset into Train and Test sets. These tests are used while calculating accuracy for the dataset and predicting the output
 - i.e. the CO₂ emitted value from the vehicle.
- Analyze the pre-processed data: involves the dropping few columns we don't require for
 - Our prediction, understanding the data type and summary of the features, observing numerical
 - and categorical values and converting into numerical values if any required.
- Train and test the machine with pre-processed data: Data is to be split into dependent and independent variables. The dependent variable is nothing but output in the dataset and the independent variable is all inputs in the dataset. The data can be divided into train and test sets by considering 80% of the data for training and 20% for testing. The split can be done by passing an argument “random state”.
- Save the model and its dependencies: Pickle is used for serializing and de- serializing Python object structures called as flattening. Serialization refers to the process of converting an object in memory to a byte stream that can be stored on disk or sent over a network. Later on, this character stream can then be retrieved and de-serialized back to a Python object.
- Build a Web application using flask that integrates with the model built.
- Static: this folder contains the images and the CSS styles for the webpage we designed.
- Templates: this folder contains the .html files we created for building our web page.
- Training: this folder contains the .ipynb file we created and trained and tested the dataset and performed data visualization techniques.
- The main folder contains the .pkl file i.e. the pickle file.

After training and testing the model, the model should be dumped and saved. This file should be placed

in the main folder.

Some of the functions used in Flask are:

- app – our flask application name
- model – it will contain the model which we build
- @app.route() – it is a decorator that can redirect to different functions.
 - i. one for routing to the home page ('/')
 - ii. route to the prediction page ('/Prediction')
 - iii. route to the same home page itself ('/home1')
 - iv. routing to the result page ('/predict')
- render_template () – it is used for render our html page from the templates folder
predict () – is taking the values form the prediction page and storing it into a variable and then we are creating a Data Frame along with the values and 9 independent features and finally we are predicting the values using or loaded model which we build and storing the output in a variable and returning it to the result page.
- app.run(debug=False) – for running our app.

The code for the Flask application is provided in the appendix.

Using the web applications, we build three html pages:

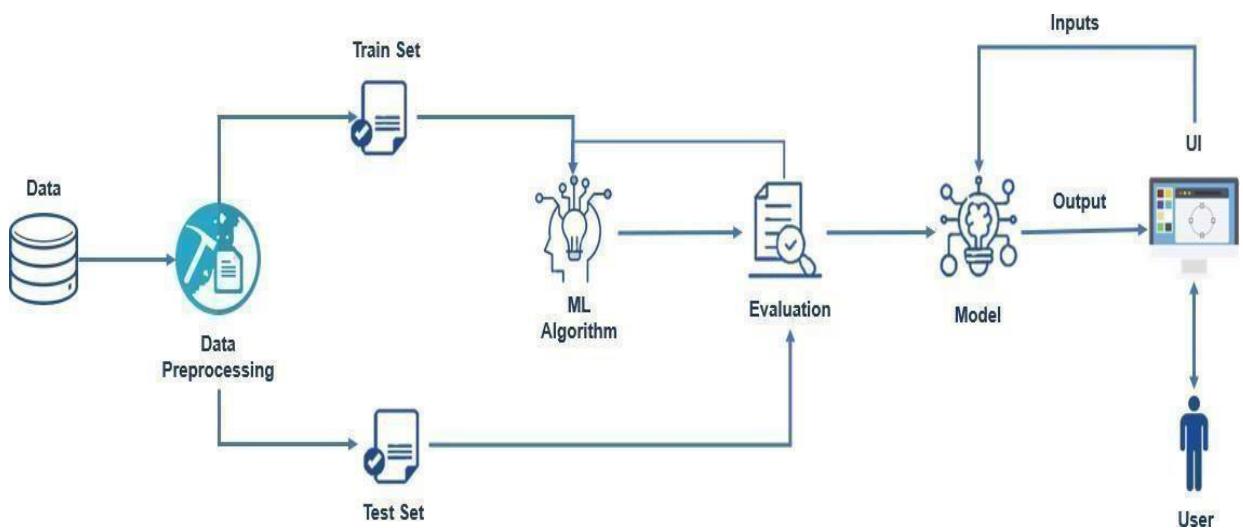
The home page: contains the introduction of the project. This page navigates to the prediction page where we give inputs and predict our model.

1. The prediction page: contains the inputs of the vehicles i.e. the features of the cars. This page navigates to the final page where the CO2 emitted value is shown.
2. The final page: contains the value of the predicted CO2 emitted by a vehicle.

By doing all these applications we can predict the amount of CO2 emitted by a vehicle and decrease the adverse effects caused by CO2 emission. This is the solution for the above discussed problem.

3. THEORITICAL ANALYSIS

BLOCK DIAGRAM:



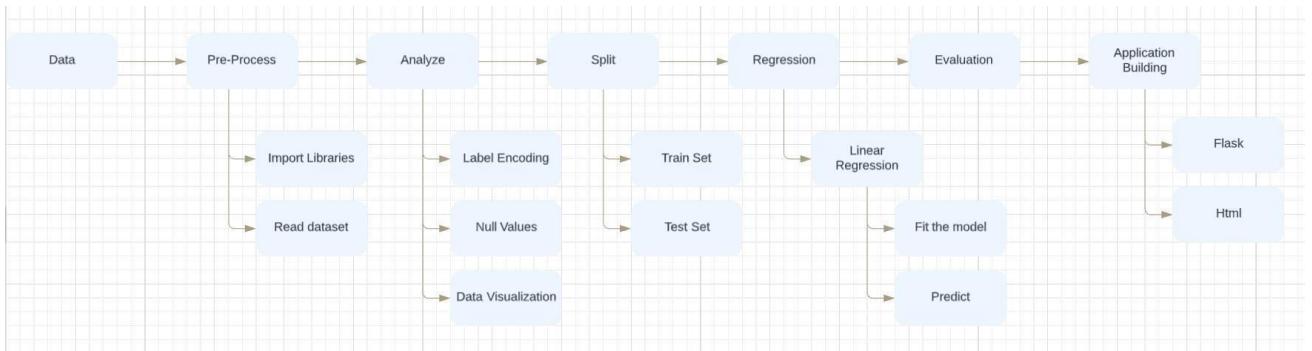
The above diagram is a representation of how the Using ML algorithm, the project works. The data is pre- processed. The data is divided into train and test sets. Using ML algorithm, the data is evaluated. The ML algorithms include regression and classification and, in this project, linear regression is used as the output is in continuous data.

The model is then used for prediction. When the user gives the inputs to the model, the model predicts the amount of CO₂ released by the vehicle and gives the output to the user. The model uses Linear Regression and fits the input train and output train into the regression.

For the output to be displayed, web applications are built using Flask. The first page will be the introduction of the project and this page navigates the user to the input pages where the user can give the features of the vehicle and predict the CO₂ emission. This page navigates the user by one click on the predict button to the final page where the user will be able to see the predicted value depicted

by the model which is obtained by the Machine Learning Algorithms.

SOFTWARE DESIGNING:



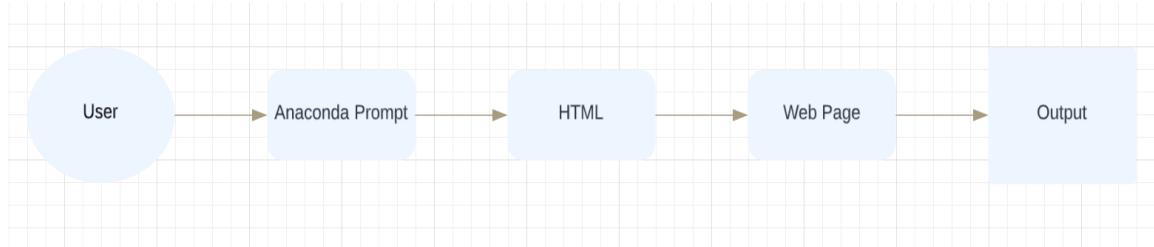
The above picture is a design of how the project works.

- The data is loaded and pre-processed.
- While pre-processing the data, the required libraries are imported, the dataset is read.
- The next step is to analyze the dataset. Analyzing the dataset includes:
 - Handling the null values i.e. checking if the dataset contains any null values or not, if contained replacing the null values with a specific value in the dataset.
 - Performing data visualization techniques like plotting graphs between the input or the output to understand the dataset. Few data visualization techniques involve bar-graph, bar-plot etc.
 - Finding the correlation between the independent variables. Correlation is a statistical relationship between two variables and it could be positive or negative. The correlation can be found by using heat map.
 - i. Positive: both variables move in the same direction
 - ii. Negative: both variables are inversely proportional i.e. when the value of one variable increases, the values of the other one decreases.
 - Label Encoding: this converts the categorical values to numerical values.
- The dataset is to be split into train and test sets. 80% of the data is trained and 20% of the data is tested.

- Next to the dataset the Machine learning algorithms are to be performed. Since the dataset is continuous dataset Regression is used. Using Linear Regression, the model is fitted.

Syntax: `variable_name = LinearRegression ()` `variable_name=variable_name.fit(x_train, y_train)`

- Once the model is trained, the model is ready to predict. We used “predict method” on the model and pass the parameter: `x_test`. The predicted value:
i.e. the output will be stored in “`y_pred`”.
- The next step in the project is to evaluate the model. This can be done by calculating:
 - Mean Absolute Error: measure of errors between paired observations expressing the same phenomenon.
 - Mean Squared Error: average of the squared difference between the target value and the value predicted by the regressionmodel.
 - Root Mean Square Error: is the square root of the averaged squared difference between the target value and the value predicted by the model.
- The last step in the design is to run the application. This is done using the Flask application. More detailed information is given in the following picture and description.



The above diagram is the representation of Flask application.

- The user accesses anaconda prompt and navigates to the folder where the files are located.
- Next is to navigate to the local host where one can view the web page.
- It runs on the localhost:5000

4. EXPERIMENTAL INVESTIGATION

Various test cases were developed for each test scenario to check the correctness of the predicted value. Manual testing was also performed for each of the test case where the inputs are given and executed without any tool. The flask application used in the project also gives the value which is similar to the manual tested value. The given table below is few features from the dataset and CO2 emission value is predicted and calculated by manual.

Test Case	Input									Expected Result	Actual Result	Status
	Type	Class	Engine Size	Cylinder Size	Transmission	Fuel Type	Fuel Consumption City	Fuel Consumption Highway	Combine Fuel			
1	Audi	Full Size	2	4	AV6	Diesel	9.9	6.7	33	173.66	173.62	Pass
2	BMW	Mini van	3.7	6	M7	Natural Gas	6	5.8	25	216.21	216.21	Pass
3	Fiat	Mid size	1.8	12	A6	Ethanol	11.2	7.5	28	265.68	265.68	Pass
4	Honda	SUV small	5.9	8	AM5	Regular Gasoline	13.4	12.6	27	299.54	299.55	Pass
5	Jeep	VAN cargo	4.7	4	AS8	Natural Gas	17.4	11.3	29	260.28	260.23	Pass
6	Kia	Mini van	5.9	10	M7	Diesel	9.9	7.4	18	283.91	283.91	Pass
7	Nissan	Full size	2	8	AS10	Ethanol	11.5	8.1	19	266.411	266.4	Pass
8	Scion	SUV small	2.4	6	AV7	Regular Gasoline	12.8	7.5	30	249.44	249.44	Pass

- The table contains input of the features and the expected and actual value of the CO2 emitted by the project. The last column judges if the actual and expected result are similar and gives a pass/fail status.
- The model predicts the CO2 value based on the inputs given.
- Pass status indicates that the expected and actual result is similar to each other.
- There are many other features of the vehicles but for instance a few are taken and the CO2 is calculated.

5. CONCLUSION

Based on the analysis of vehicle features, and by the proximate analysis data to predict fuel characteristic factor, the calculation method of predicting CO₂ emission of vehicles is established. Successfully we were able to predict the CO₂ Emissions for the dataset with 86% accuracy by calculating the r₂_score of the dataset.

Prediction with high accuracy can give information concerning about CO₂ emissions. The available data can be used to extract the features, know the behavior of cars, and try to reduce the emissions. The model is able to extract the CO₂ emitted value by all the features and also is able to generate an error when the user tries to give a null value as an input.

6. FUTURE SCOPE

UG Project Phase-2 is the extension of UG Project Phase-1. UG Project Phase-2 involves all the coding and implementation of the design which we have retrieved from UG Project Phase-1. All the implementation is done and conclusions will be retrieved in the phase. We will also work on the applications, advantages and disadvantages of the project in this phase. Future scope of the project will be also discussed in the UG Project Phase-1.

PREDICTING CO2 EMISSION IN VEHICLES USING MACHINE LEARNING

A UG PROJECT PHASE-2 REPORT

Submitted to

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
HYDERABAD**

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

Submitted by

AMARAVADI JAVALI	19UK1A05J1
LINGALA SHALINI	19UK1A05J8
KATABATHINI BHARATH	19UK1A05L2
JAMPALA SAITEJA	19UK1A05J3

Under the esteemed guidance of

Mr. T. SANATH KUMAR

(Assistant Professor)



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE**

(Affiliated to JNTUH, Hyderabad)

Bollikunta, Warangal-506005

2019– 2023

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
VAAGDEVI ENGINEERING COLLEGE
BOLLIKUNTA, WARANGAL – 506005
2019 – 2023**



**CERTIFICATE OF COMPLETION OF
A UG PROJECT PHASE-2**

This is to certify that the Major project entitled “**PREDICTING CO2 EMISSION IN VEHICLES USING MACHINE LEARNING**” is being submitted by **A.JAVALI (19UK1A05J1),L.SHALINI (19UK1A05J8),K.BHARATH (19UK1A05L2),J.SAITEJA (19UK1A05J3)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science and Engineering to Jawaharlal Nehru Technological University Hyderabad** during the academic year **2022-23**, is a record of work carried out by them under the guidance and supervision.

Project Guide

Mr. T. SANATH KUMAR
(Assistant Professor)

Head of the Department

Dr. R. NAVEEN KUMAR
(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. P. PRASAD RAO**, Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-2 in the institute.

We extend our heartfelt thanks to **Dr. R. NAVNEEN KUMAR**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-2.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-I and for their support in completing the UG Project Phase-2.

We express heartfelt thanks to the guide, **Mr. T. SANATH KUMAR** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-2.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

A.JAVALI **(19UK1A05J1)**

L.SHALINI **(19UK1A05J8)**

K.BHARATH **(19UK1A05L2)**

J.SAITEJA **(19UK1A05J3)**

TABLE OF CONTENTS:

1. INTRODUCTION.....
2. CODE SNIPPETS.....
2.1 MODEL CODE.....
2.2 HTML CODE AND PYTHON CODE.....
3. CONCLUSION.....
4. APPLICATIONS.....
5. ADVANTAGES.....
6. DISADVANTAGES.....
7. FUTURE SCOPE.....
8. BIBLIOGRAPHY.....
9. HELP FILE.....

LIST OF FIGURES

-PAGE NO

- Figure 1:** .ipynb code describing importing libraries and displaying few rows from the dataset.....
- Figure 2:** .ipynb code describing drop() and info() methods.....
- Figure 3:** .ipynb code describing info() and describe() methods.....
- Figure 4:** .ipynb using unique() finding the data type.....
- Figure 5:** .ipynb code describes handling of continuous variable data.....
- Figure 6:** .ipynb code describing fuel type and index values.....
- Figure 7:** .ipynb code describes import label encoder.....
- Figure 8:** .ipynb code describing sorting values.....
- Figure 9:** .ipynb code describing about bar graph.....
- Figure 10:** .ipynb describing corr() and subplots() methods.....
- Figure 11:** .ipynb describing output of subplot() method.....
- Figure 11:** .ipynb describing about the sort values in ascending order.....
- Figure 11:** .ipynb code describing of splitting data into train and test variables, linear regression, and predicting x values.....
- Figure 11:** .ipynb describes the output of predicting values.....
- Figure 11:** .ipynb code describes importing metrics, predicting y values.....
- Figure 11:** .py code used for rendering all the HTML pages.....
- Figure 11:** home.html page is the code for home page of our Web Application.....
- Figure 11:** result.html is the page which displays the result that emission of CO2 releases from the vehicles.....
- Figure 11:** home page.....
- Figure 11:** input page.....
- Figure 11:** prediction(which type of car).....
- Figure 11:** output.....

1. INTRODUCTION

With the development of the world economy, the situation of carbon dioxide emissions control is becoming increasingly serious. CO₂ emissions are the primary driver of the global climate change. It is widely recognized to avoid the worst impacts of climate change. A typical vehicle emits about 4.6 metric tons of carbon dioxide per year.

This causes global warming and effects the environment. The number can vary based on a vehicle's fuel, fuel economy and the number of miles driven per year. When gasoline in the vehicle burns, the carbon and the hydrogen separate.

The hydrogen combines with oxygen to form water (H₂O), and carbon combines with oxygen to form carbon dioxide (CO₂). This is how CO₂ is emitted from vehicles.

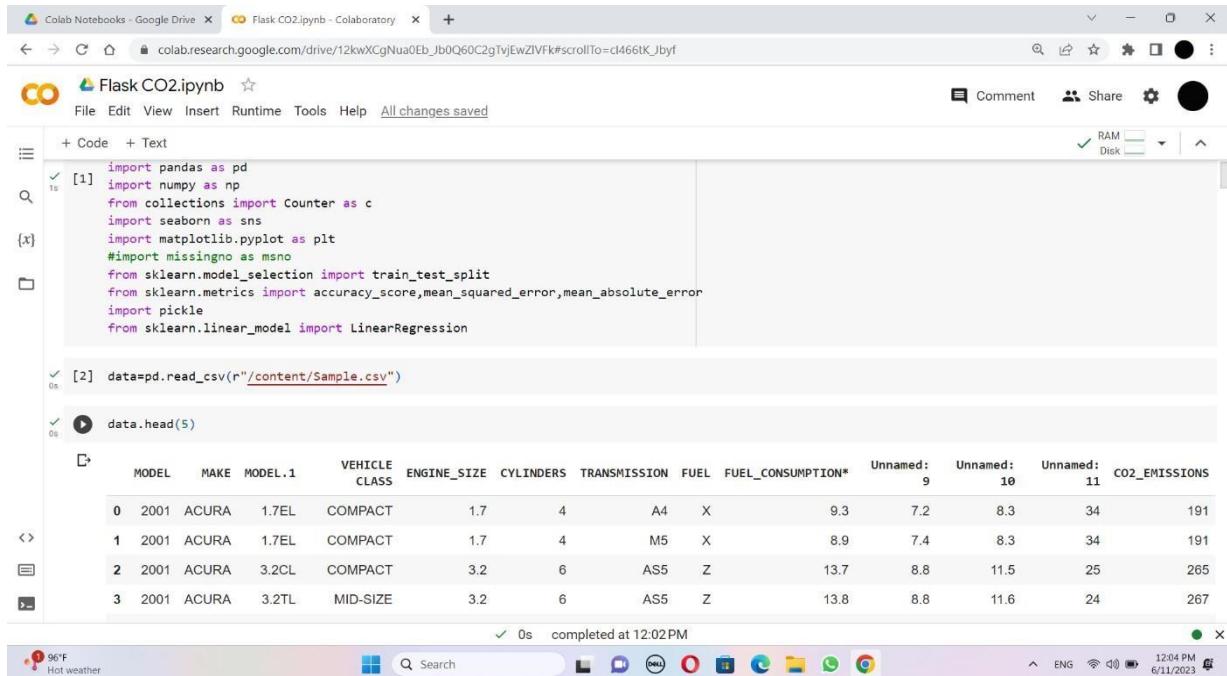
This project will be predicting the amount of CO₂ released by the vehicles by exploring the inputs of the vehicles such as engine-size, fuel-consumption etc.

The goal of the project is to develop an emission predictor which predicts the amount of the CO₂ emitted by a vehicle with certain characteristics. The model will be trained and tested using Machine Learning Techniques.

The predictor is built by using Flask applications. HTML and CSS are used for creating and designing the webpage.

2. CODE SNIPPETS

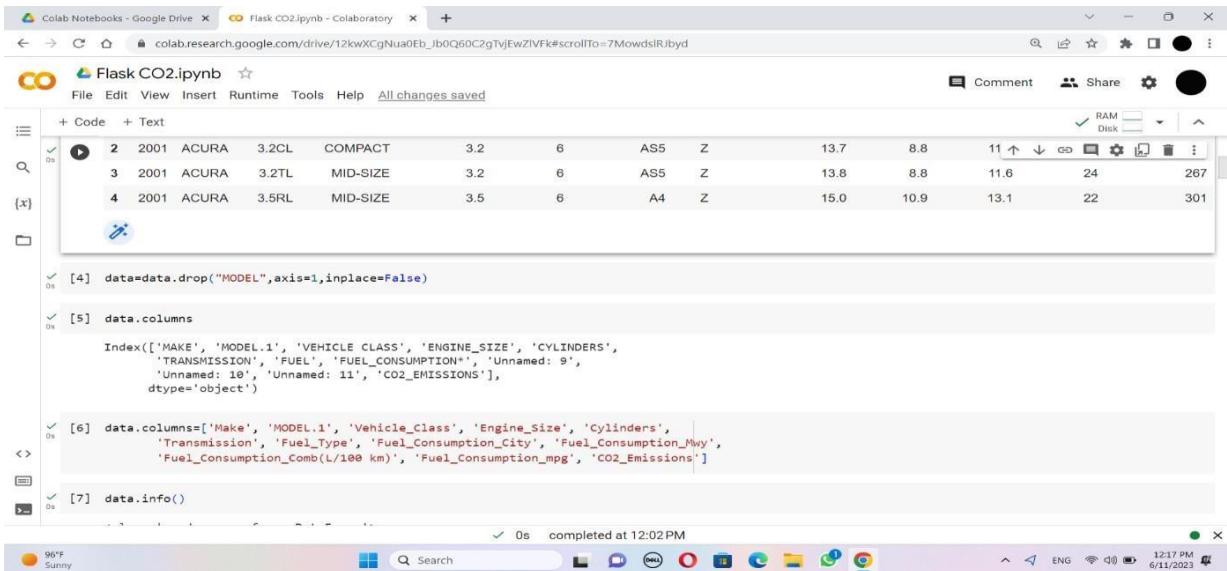
MODEL CODE



The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell [1] imports various Python libraries including pandas, numpy, Counter, seaborn, matplotlib.pyplot, missingno, train_test_split, accuracy_score, mean_squared_error, mean_absolute_error, pickle, and LinearRegression from sklearn.linear_model. The code cell [2] reads a CSV file named "Sample.csv" into a DataFrame named "data". A preview of the first five rows of the dataset is shown:

	MODEL	MAKE	MODEL.1	VEHICLE CLASS	ENGINE_SIZE	CYLINDERS	TRANSMISSION	FUEL	FUEL_CONSUMPTION*	Unnamed: 9	Unnamed: 10	Unnamed: 11	CO2_EMISSIONS
0	2001	ACURA	1.7EL	COMPACT	1.7	4	A4	X	9.3	7.2	8.3	34	191
1	2001	ACURA	1.7EL	COMPACT	1.7	4	M5	X	8.9	7.4	8.3	34	191
2	2001	ACURA	3.2CL	COMPACT	3.2	6	AS5	Z	13.7	8.8	11.5	25	265
3	2001	ACURA	3.2TL	MID-SIZE	3.2	6	AS5	Z	13.8	8.8	11.6	24	267

Figure 1: .ipynb code describing importing libraries and displaying the few rows from the dataset.



The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell [4] drops the "MODEL" column from the DataFrame "data" using the drop() method with axis=1 and inplace=False. The code cell [5] displays the columns of the DataFrame "data". The code cell [6] renames the columns to "Make", "Model.1", "Vehicle_Class", "Engine_Size", "Cylinders", "Transmission", "Fuel_Type", "Fuel_Consumption_City", "Fuel_Consumption_Hwy", "Fuel_Consumption_Comb(L/100 km)", "Fuel_Consumption_mpg", and "CO2_Emissions". The code cell [7] displays the information about the DataFrame "data" using the info() method.

Figure 2: .ipynb code describing drop() and info() methods.

```

[7]: <class 'pandas.core.frame.DataFrame'>
RangeIndex: 679 entries, 0 to 678
Data columns (total 12 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Make              679 non-null    object  
 1   MODEL.1          679 non-null    object  
 2   Vehicle_Class    679 non-null    object  
 3   Engine_Size      679 non-null    float64 
 4   Cylinders        679 non-null    int64  
 5   Transmission     679 non-null    object  
 6   Fuel_Type         679 non-null    object  
 7   Fuel_Consumption_City 679 non-null    float64 
 8   Fuel_Consumption_Mwy   679 non-null    float64 
 9   Fuel_Consumption_Comb(L/100 km) 679 non-null    float64 
 10  Fuel_Consumption_mpg    679 non-null    int64  
 11  CO2_Emissions      679 non-null    int64  
dtypes: float64(4), int64(3), object(5)
memory usage: 63.8+ KB

[8]: data.describe()

   Engine_Size  Cylinders  Fuel_Consumption_City  Fuel_Consumption_Mwy  Fuel_Consumption_Comb(L/100 km)  Fuel_Consumption_mpg  CO2_Emissions
count    679.000000  679.000000       679.000000       679.000000       679.000000       679.000000       679.000000
mean     3.252577   5.798233      14.591900      10.613844      12.802798      23.107511      293.656848
std      1.203751   1.531073      3.025654      2.357724      2.685590      5.308083      60.372456

```

Figure 3: .ipynb code describing info() and describe() methods.

```

[8]: min    1.000000  3.000000  4.900000  4.000000  4.500000  14.000000  104.000000
      25%   2.200000  4.000000  12.700000  9.000000  11.000000  19.000000  253.000000
      50%   3.000000  6.000000  14.300000  10.100000  12.300000  23.000000  283.000000
      75%   4.200000  6.000000  16.650000  12.500000  14.850000  26.000000  340.000000
      max   8.000000  12.000000  23.200000  18.000000  20.800000  63.000000  478.000000

[9]: np.unique(data.dtypes,return_counts=True)

(array(['int64'], dtype='float64'), dtype='O'), dtype=object),
array([3, 4, 5])

cat=data.dtypes[data.dtypes=='O'].index.values
cat

array(['Make', 'MODEL.1', 'Vehicle_Class', 'Transmission', 'Fuel_Type'],
      dtype=object)

[11]: for i in cat:
        print("Column : ", i)
        print("count of classes : ", data[i].nunique())
        print(c(data[i]))
        print('*'*120)

```

Figure 4: .ipynb using unique() finding the data type.

The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell contains several Counter objects for different vehicle features:

```

Column : Make
count of classes : 34
Counter({'FORD': 63, 'CHEVROLET': 68, 'DODGE': 47, 'BMW': 43, 'GMC': 41, 'TOYOTA': 36, 'VOLKSWAGEN': 35, 'MERCEDES-BENZ': 29, 'CHRYSLER': 25, 'AUDI': 2
*****
Column : MODEL.1
count of classes : 351
Counter({'RAM 1500': 9, 'DAKOTA': 6, 'DAKOTA 4X4': 6, 'K1500 SIERRA 4X4': 6, 'JETTA': 6, 'C1500 SILVERADO': 5, 'K1500 SILVERADO 4X4': 5, 'F150': 5, 'F1
*****
Column : Vehicle_Class
count of classes : 14
Counter({'COMPACT': 145, 'PICKUP TRUCK - STANDARD': 101, 'SUV': 98, 'MID-SIZE': 82, 'SUBCOMPACT': 62, 'TWO-SEATER': 31, 'STATION WAGON - MID-SIZE': 31,
*****
Column : Transmission
count of classes : 8
Counter({'A4': 312, 'M5': 198, 'A5': 106, 'M6': 26, 'AS5': 15, 'AS4': 14, 'A3': 7, 'AV': 1})
*****
Column : Fuel_Type
count of classes : 5
Counter({'X': 442, 'Z': 223, 'D': 6, 'E': 4, 'N': 4})
*****
```

Cell [12] contains code for handling transmission types:

```
[12] data["Transmission"] = np.where(data["Transmission"].isin(["A4", "A5", "A3"]), "Automatic", data["Transmission"])
data["Transmission"] = np.where(data["Transmission"].isin(["M5", "M6"]), "Manual", data["Transmission"])
data["Transmission"] = np.where(data["Transmission"].isin(["AS4", "AS5"]), "Automatic with Select Shift", data["Transmission"])
data["Transmission"] = np.where(data["Transmission"].isin(["AV"]), "Continuously Variable", data["Transmission"])
c(data["Transmission"])
```

The status bar at the bottom shows "96°F Sunny" and the date "6/11/2023".

Figure 5: .ipynb code describes handing of continuous variable data.

The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell contains several Counter objects for different fuel types:

```

[12] Counter({'Automatic': 425,
'Manual': 224,
'Automatic with Select Shift': 29,
'Continuously Variable': 1})
```

Cell [13] contains code for handling fuel types:

```
[13] data["Fuel_Type"] = np.where(data["Fuel_Type"]=="Z", "Premium Gasoline", data["Fuel_Type"])
data["Fuel_Type"] = np.where(data["Fuel_Type"]=="X", "Regular Gasoline", data["Fuel_Type"])
data["Fuel_Type"] = np.where(data["Fuel_Type"]=="D", "Diesel", data["Fuel_Type"])
data["Fuel_Type"] = np.where(data["Fuel_Type"]=="E", "Ethanol(EBS)", data["Fuel_Type"])
data["Fuel_Type"] = np.where(data["Fuel_Type"]=="N", "Natural Gas", data["Fuel_Type"])
c(data["Fuel_Type"])
```

Cell [14] contains code for getting index values of non-integer columns:

```
[14] data.dtypes[data.dtypes!='O'].index.values
array(['Engine_Size', 'Cylinders', 'Fuel_Consumption_City',
'Fuel_Consumption_Mwy', 'Fuel_Consumption_Comb(L/100 km)',
'Fuel_Consumption_mpg', 'CO2_Emissions'], dtype=object)
```

Cell [15] contains code for summing null values:

```
[15] data.isnull().sum()
```

The status bar at the bottom shows "96°F Sunny" and the date "6/11/2023".

Figure 6: .ipynb code describing fuel type and index values.

```

[15]: Make          0
      MODEL_1        0
      Vehicle_Class  0
      Engine_Size    0
      Cylinders     0
      Transmission   0
      Fuel_Type      0
      Fuel_Consumption_City 0
      Fuel_Consumption_Mwy 0
      Fuel_Consumption_Comb(L/100 km) 0
      Fuel_Consumption_mpg 0
      CO2_Emissions   0
      dtype: int64

from sklearn.preprocessing import LabelEncoder
x='*'
for i in cat:
    print("LABEL ENCODING OF:",i)
    LE=LabelEncoder()
    print(c(data[i]))
    data[i]=LE.fit_transform(data[i])
    print(c(data[i]))
    print(x*100)

LABEL ENCODING OF: Make
Counter({'FORD': 63, 'CHEVROLET': 60, 'DODGE': 47, 'BMW': 43, 'GMC': 41, 'TOYOTA': 36, 'VOLKSWAGEN': 35, 'MERCEDES-BENZ': 29, 'CHRYSLER': 25, 'AUDI': 2
Counter({9: 63, 5: 60, 8: 47, 2: 43, 10: 41, 31: 36, 32: 35, 21: 29, 6: 25, 1: 24, 20: 24, 33: 24, 22: 23, 25: 19, 27: 15, 28: 15, 30: 15, 11: 14, 29:

```

Figure 7: .ipynb code describes import label encoder.

```

LABEL ENCODING OF: Make
Counter({'FORD': 63, 'CHEVROLET': 60, 'DODGE': 47, 'BMW': 43, 'GMC': 41, 'TOYOTA': 36, 'VOLKSWAGEN': 35, 'MERCEDES-BENZ': 29, 'CHRYSLER': 25, 'AUDI': 2
Counter({9: 63, 5: 60, 8: 47, 2: 43, 10: 41, 31: 36, 32: 35, 21: 29, 6: 25, 1: 24, 20: 24, 33: 24, 22: 23, 25: 19, 27: 15, 28: 15, 30: 15, 11: 14, 29:
*****+
LABEL ENCODING OF: MODEL_1
Counter({'RAM 1500': 9, 'DAKOTA': 6, 'DAKOTA 4X4': 6, 'K1500 SIERRA 4X4': 6, 'JETTA': 6, 'C1500 SILVERADO': 5, 'K1500 SILVERADO 4X4': 5, 'F150': 5, 'F1
Counter({239: 9, 101: 6, 102: 6, 175: 6, 171: 6, 65: 5, 176: 5, 130: 5, 131: 5, 64: 5, 34: 4, 37: 4, 349: 4, 73: 4, 79: 4, 262: 4, 168: 4, 281: 4, 183:
*****+
LABEL ENCODING OF: Vehicle_Class
Counter({'COMPACT': 145, 'PICKUP TRUCK - STANDARD': 101, 'SUV': 98, 'MID-SIZE': 82, 'SUBCOMPACT': 62, 'TWO-SEATER': 31, 'STATION WAGON - MID-SIZE': 31,
Counter({0: 145, 6: 101, 10: 98, 2: 82, 9: 62, 11: 31, 7: 31, 1: 30, 12: 23, 13: 18, 3: 17, 4: 17, 8: 12, 5: 12})
*****+
LABEL ENCODING OF: Transmission
Counter({'Automatic': 425, 'Manual': 224, 'Automatic with Select Shift': 29, 'Continuously Variable': 1})
Counter({0: 425, 3: 224, 1: 29, 2: 1})
*****+
LABEL ENCODING OF: Fuel_Type
Counter({'Regular Gasoline': 442, 'Premium Gasoline': 223, 'Diesel': 6, 'Ethanol(EBS)': 4, 'Natural Gas': 4})
Counter({4: 442, 3: 223, 0: 6, 1: 4, 2: 4})
*****+
[17]: MCO2=data.groupby(['Make'])['CO2_Emissions'].mean().sort_values()[:20].reset_index()
plt.figure(figsize=(25,6))
sns.barplot(x="Make",y="CO2_Emissions",data=MCO2)

<Axes: xlabel='Make', ylabel='CO2_Emissions'>
300

```

Figure 8: .ipynb code describing sorting values.

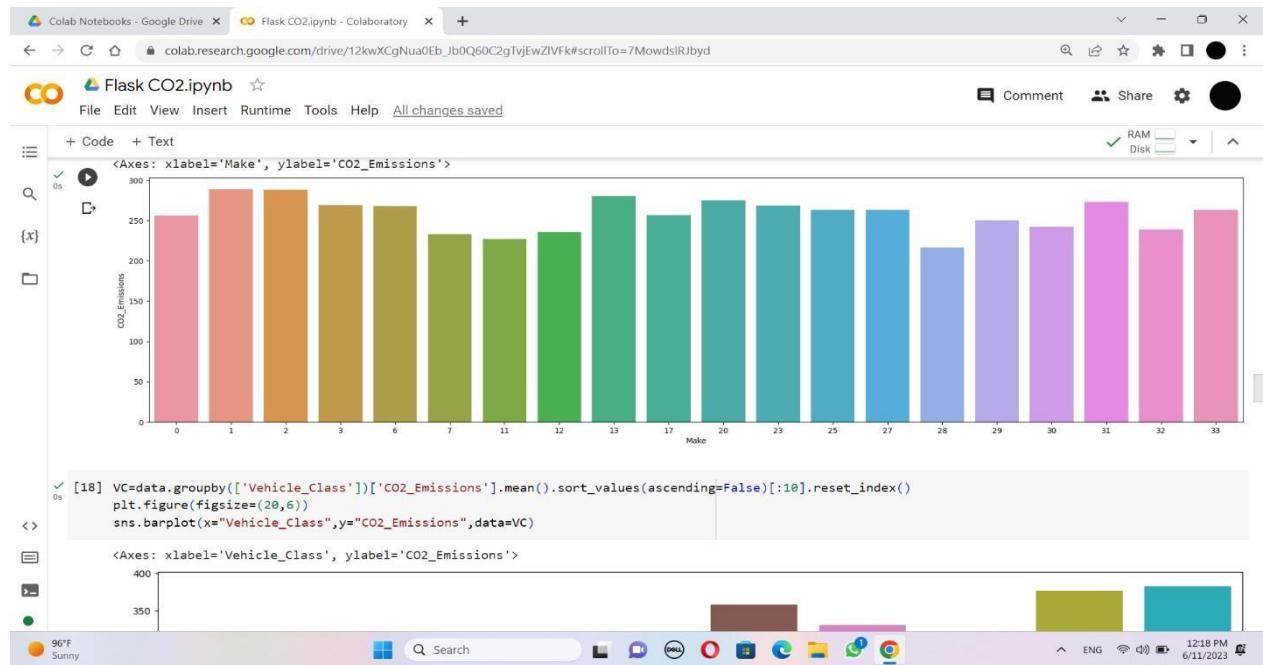


Figure 9: .ipynb describing about bar graph.

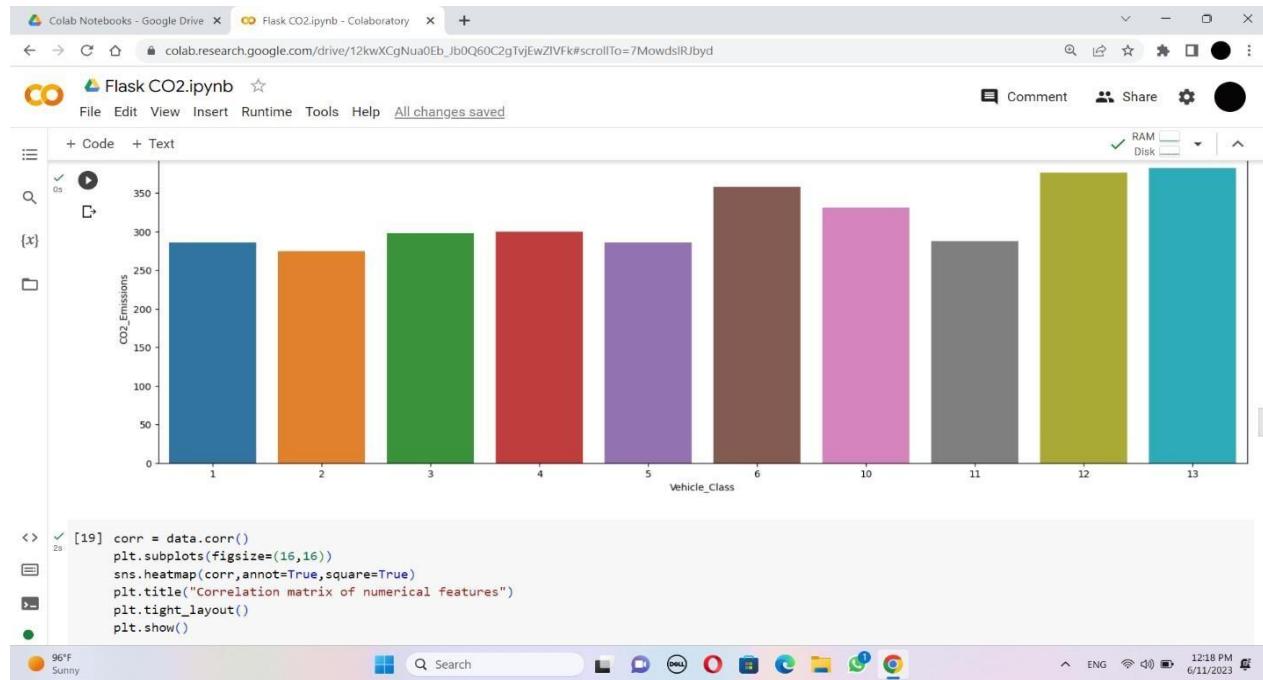


Figure 10: .ipynb describing corr() and subplots() methods.

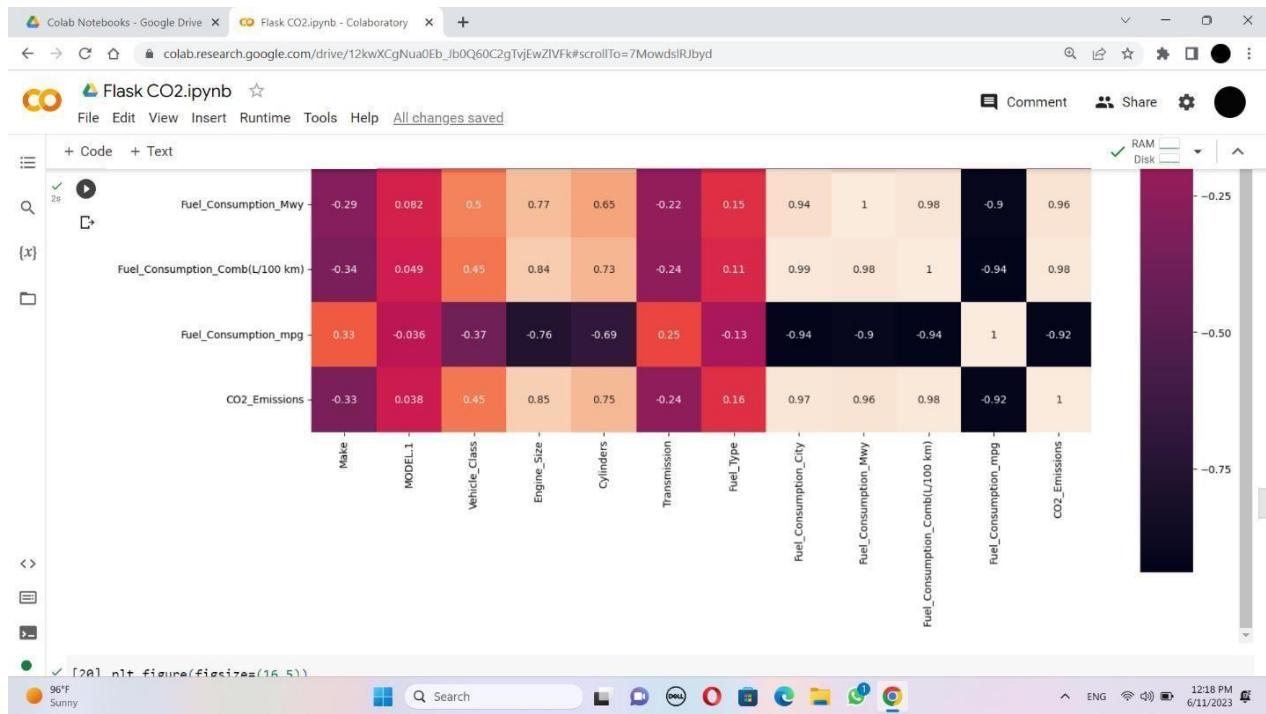
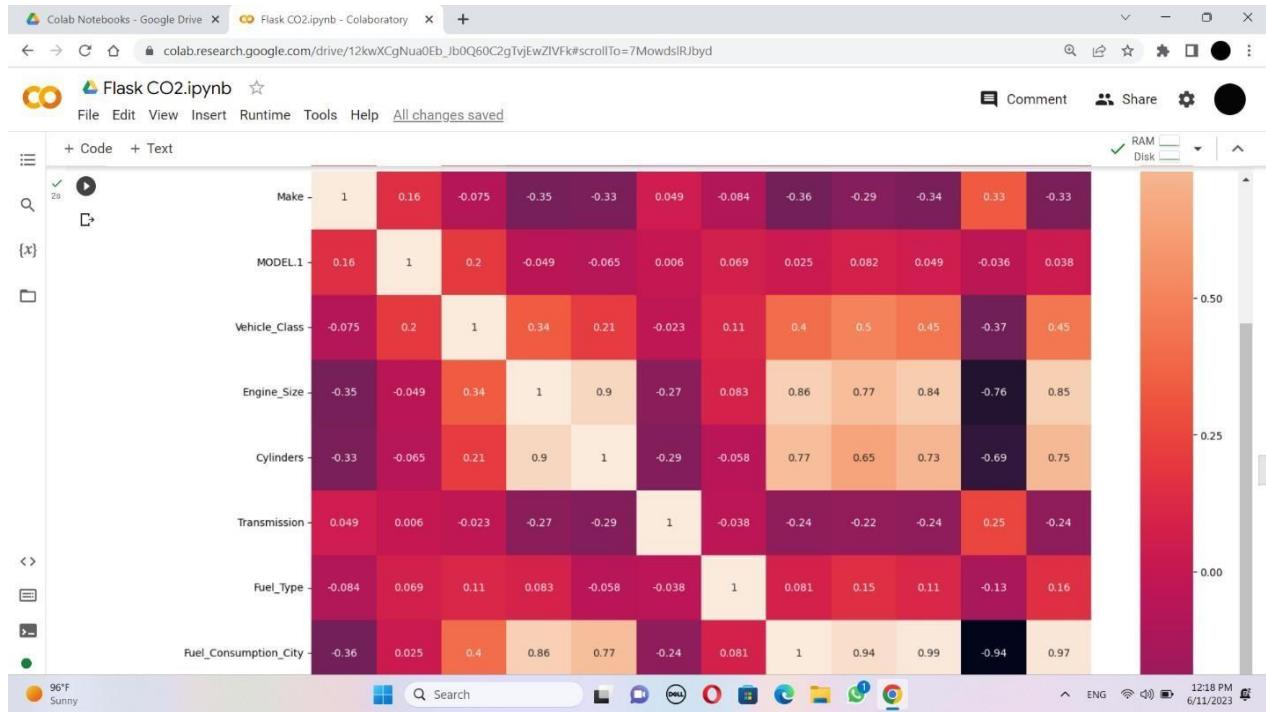


Figure 11: .ipynb describing output of subplot() method.

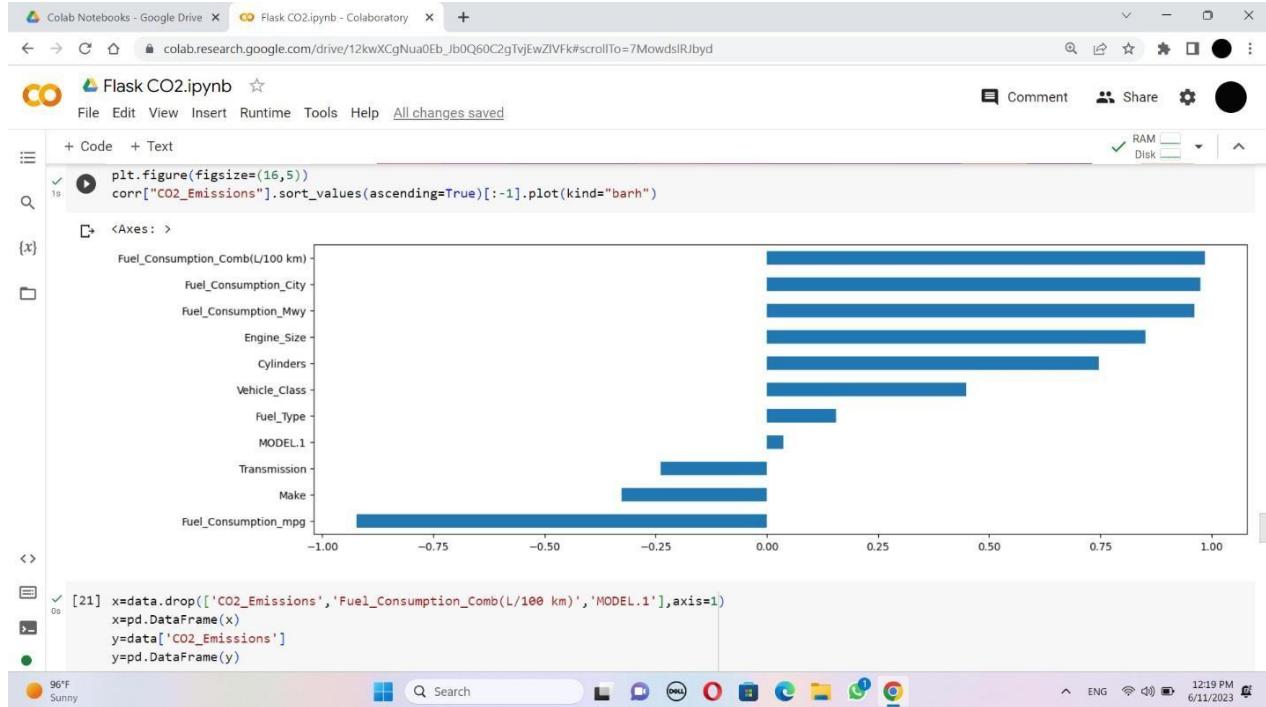


Figure 12: .ipynb describing about the sort values in ascending order.

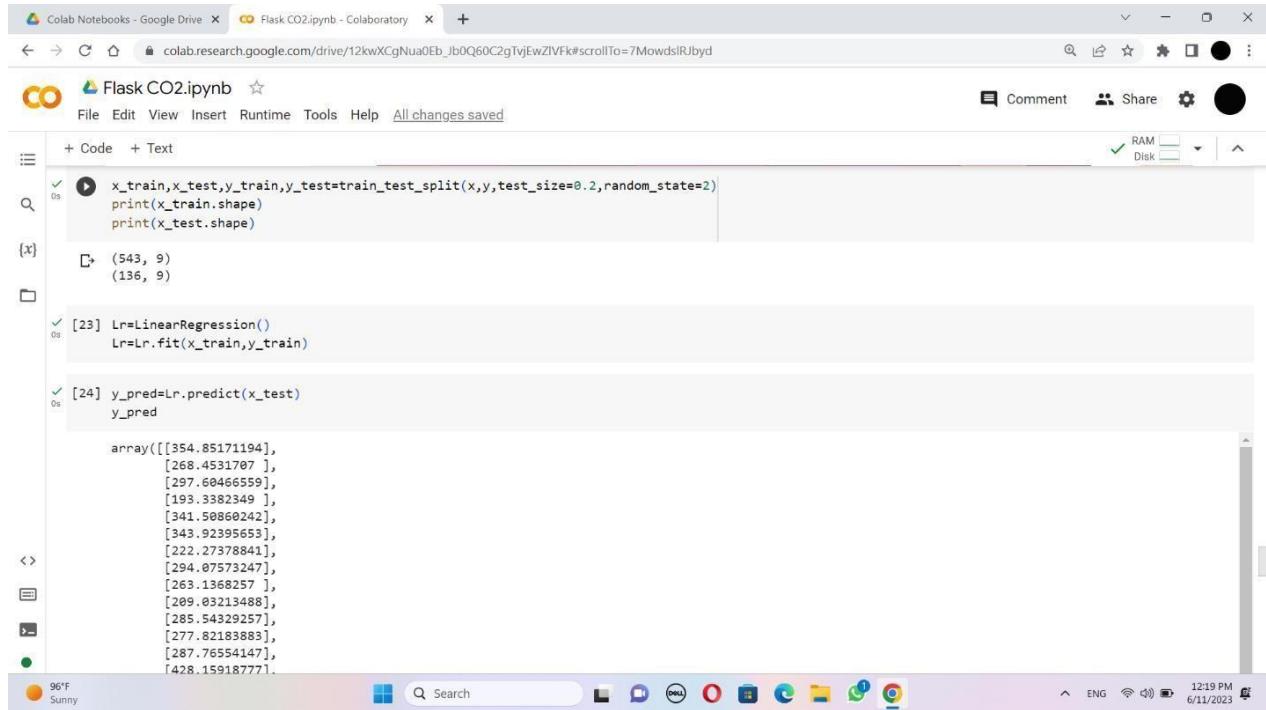


Figure 13: .ipynb code describing of splitting data into train and test variables, linear regression, and predicting x values.

The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell contains a list of numerical values:

```
[217.86410296],  
[343.0229313],  
[339.376677],  
[347.35915488],  
[232.63908778],  
[336.53765593],  
[256.38284942],  
[365.34981444],  
[348.4734845],  
[369.40493529],  
[240.17803987],  
[160.19361701],  
[266.35988375],  
[192.76228697],  
[298.09905274],  
[309.2563186],  
[389.16961768],  
[260.23251057],  
[263.669606713],  
[279.69974172],  
[282.36328002],  
[235.88174772],  
[269.37956017],  
[235.80954707],  
[397.71787918],  
[321.76032239],  
[297.18887096],  
[280.38247127],  
[199.73208844],  
[268.08133507],
```

Figure 14: .ipynb describes the output of predicting values

The screenshot shows a Google Colab notebook titled "Flask CO2.ipynb". The code cell contains the following Python code:

```
from sklearn import metrics  
print("MAE: ",metrics.mean_absolute_error(y_test,y_pred))  
print("MSE: ",metrics.mean_squared_error(y_test,y_pred))  
print("RMSE: ",np.sqrt(metrics.mean_squared_error(y_test,y_pred)))
```

Output:

```
MAE: 5.327868570518979  
MSE: 260.82431344914187  
RMSE: 16.15005143838693
```

Code cell 26:

```
[26] y=Lr.predict([[0,0,1,7,4,0,4,9,3,7,2,34])  
y
```

Output:

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names array([[194.30811772]])
```

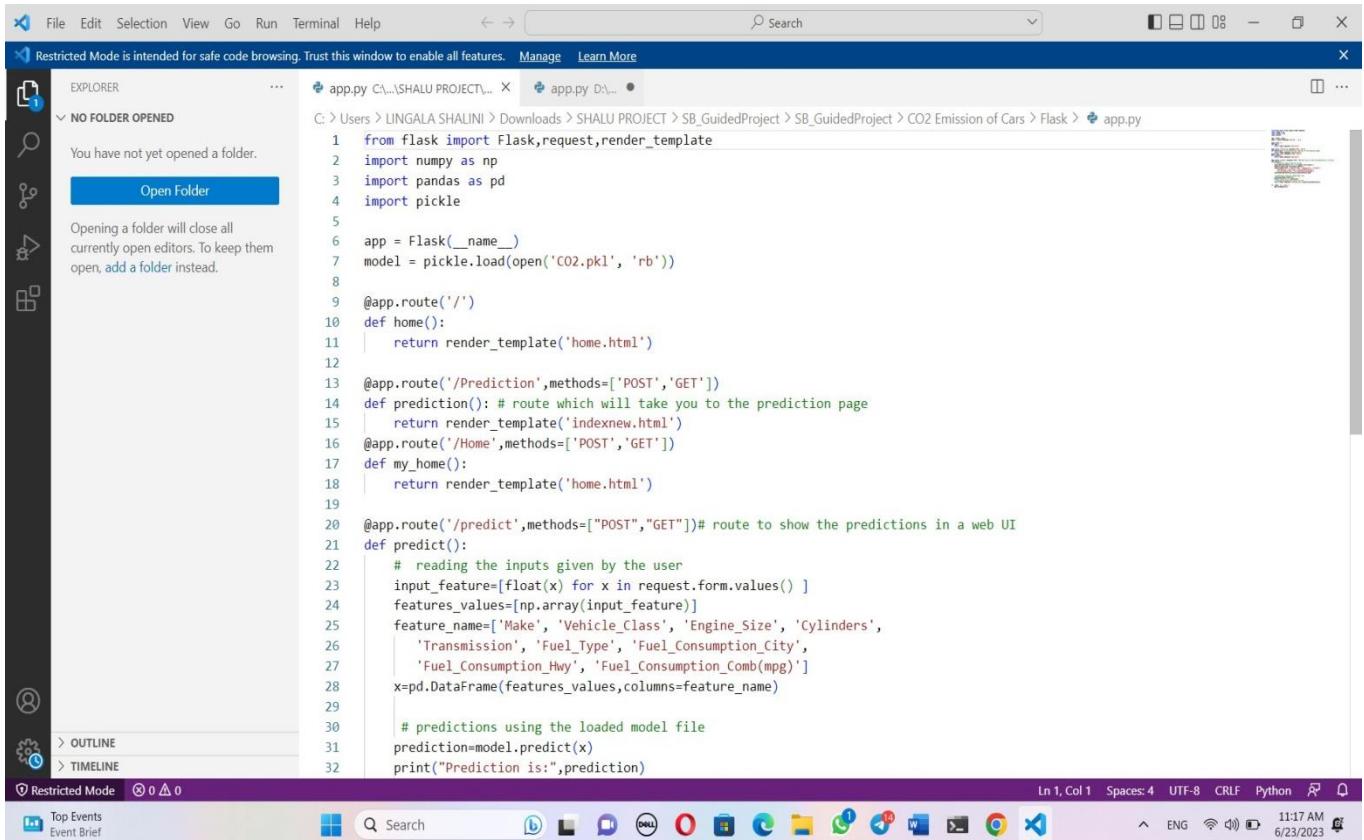
Code cell 27:

```
[27] pickle.dump(Lr,open("CO2.pkl","wb"))
```

Figure 16: .ipynb code describes importing metrics, predicting y values.

2. HTML CODE AND PYTHON CODE

1.app.py code:

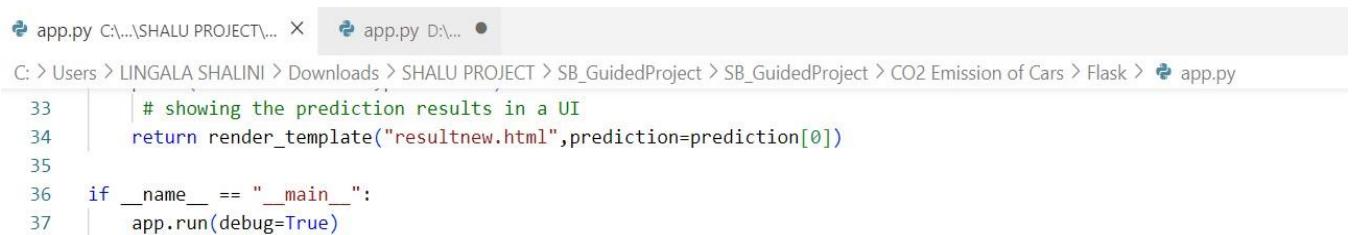


The screenshot shows a code editor window with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** Search.
- Message Bar:** Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More.
- Explorer Bar:** NO FOLDER OPENED. You have not yet opened a folder. Open Folder.
- Code Area:** The code is written in Python using the Flask framework. It includes routes for home, prediction, and a custom route. It also contains a predict function that reads user inputs and makes predictions using a loaded model.

```
1  from flask import Flask,request,render_template
2  import numpy as np
3  import pandas as pd
4  import pickle
5
6  app = Flask(__name__)
7  model = pickle.load(open('CO2.pkl', 'rb'))
8
9  @app.route('/')
10 def home():
11     return render_template('home.html')
12
13 @app.route('/Prediction',methods=['POST','GET'])
14 def prediction(): # route which will take you to the prediction page
15     return render_template('indexnew.html')
16 @app.route('/Home',methods=['POST','GET'])
17 def my_home():
18     return render_template('home.html')
19
20 @app.route('/predict',methods=["POST","GET"])# route to show the predictions in a web UI
21 def predict():
22     # reading the inputs given by the user
23     input_feature=[float(x) for x in request.form.values()]
24     features_values=[np.array(input_feature)]
25     feature_name=['Make', 'Vehicle_Class', 'Engine_Size', 'Cylinders',
26                   'Transmission', 'Fuel_Type', 'Fuel_Consumption_City',
27                   'Fuel_Consumption_Hwy', 'Fuel_Consumption_Comb(mpg)']
28     x=pd.DataFrame(features_values,columns=feature_name)
29
30     # predictions using the loaded model file
31     prediction=model.predict(x)
32     print("Prediction is:",prediction)
```

- Status Bar:** Ln 1, Col 1 Spaces: 4 UTF-8 CRLF Python
- Taskbar:** Top Events Event Brief, Search, various icons for system and application tasks, battery level, and network status.
- System Bar:** 11:17 AM 6/23/2023.



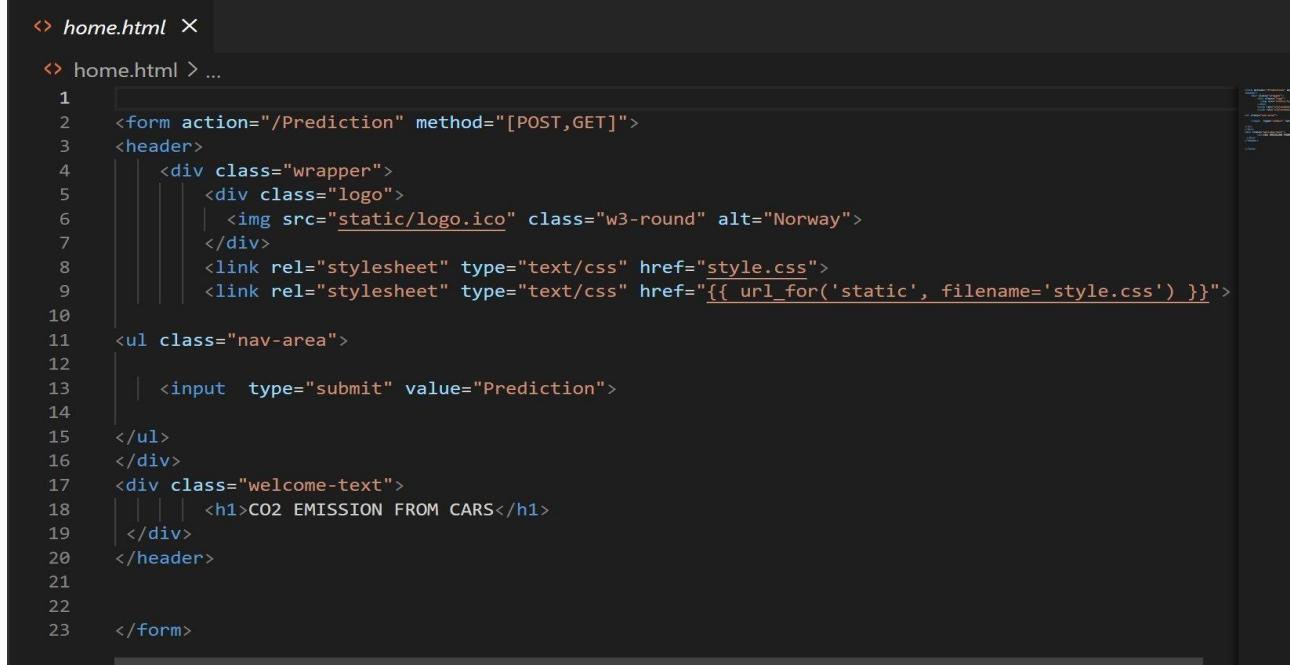
The screenshot shows a code editor window with the following details:

- File Bar:** File, Edit, Selection, View, Go, Run, Terminal, Help.
- Search Bar:** Search.
- Code Area:** The code is identical to the one in the previous screenshot, showing the prediction logic and main application run.

```
33     # showing the prediction results in a UI
34     return render_template("resultnew.html",prediction=prediction[0])
35
36 if __name__ == "__main__":
37     app.run(debug=True)
```

Figure 17: .python code used for rendering all the HTML pages.

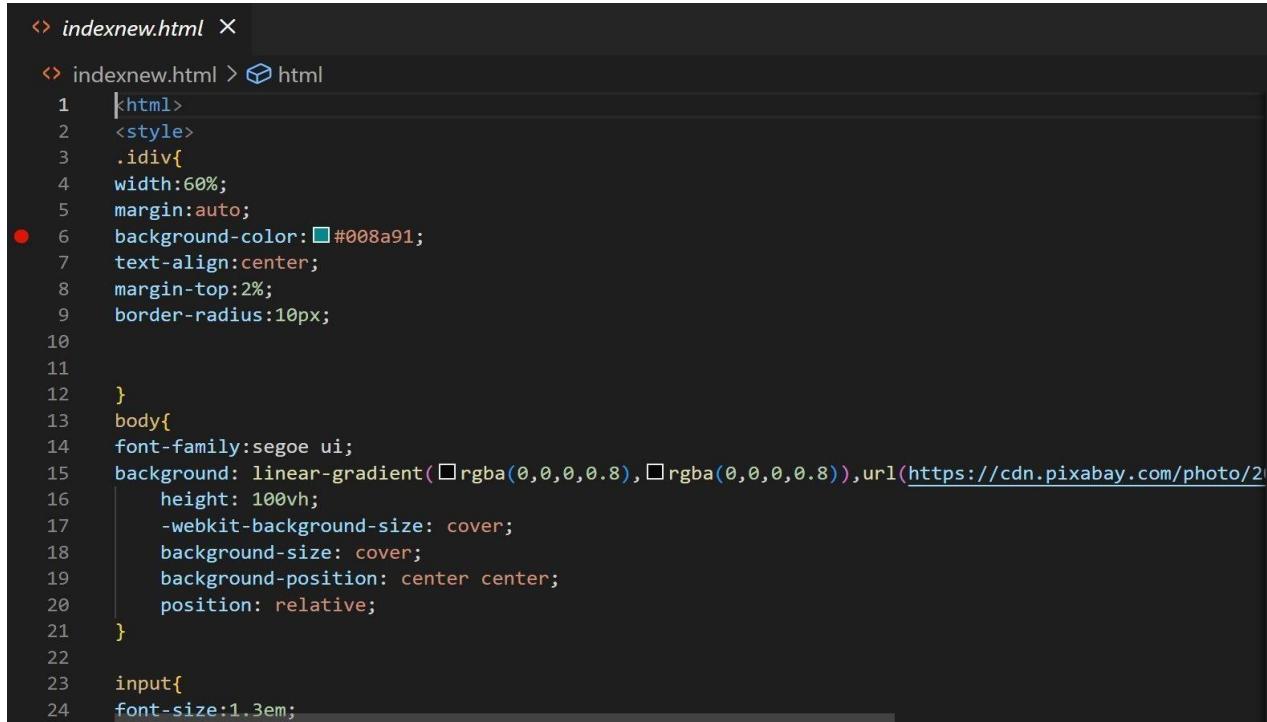
2. home.html



```
<> home.html <...>
<> home.html > ...
1  <form action="/Prediction" method="[POST,GET]">
2    <header>
3      <div class="wrapper">
4        <div class="logo">
5          | 
6        </div>
7        <link rel="stylesheet" type="text/css" href="style.css">
8        <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='style.css') }}>
9
10   <ul class="nav-area">
11     | <input type="submit" value="Prediction">
12   </ul>
13 </div>
14 <div class="welcome-text">
15   | | | <h1>CO2 EMISSION FROM CARS</h1>
16 </div>
17 </header>
18
19 </form>
```

Figure 18: home.html page is the code for home page of our Web Application.

2.index.html:



```
<> indexnew.html <...>
<> indexnew.html > ⚡ html
1   <html>
2     <style>
3       .idiv{
4         width:60%;
5         margin:auto;
6         background-color:#008a91;
7         text-align:center;
8         margin-top:2%;
9         border-radius:10px;
10
11
12     }
13     body{
14       font-family:segoe ui;
15       background: linear-gradient(rgba(0,0,0,0.8),rgba(0,0,0,0.8)),url("https://cdn.pixabay.com/photo/2016/08/08/05/43/city-1590750_960_720.jpg");
16         height: 100vh;
17         -webkit-background-size: cover;
18         background-size: cover;
19         background-position: center center;
20         position: relative;
21     }
22
23     input{
24       font-size:1.3em;
```

```
25    width:80%;  
26    text-align:center;  
27    border-color: black white;  
28  
29  
30    }  
31    ::placeholder { /* Chrome, Firefox, Opera, Safari 10.1+ */  
32    | color: olive;  
33    | opacity: 1; /* Firefox */  
34    }  
35    input placeholder{  
36    text-align:center;  
37    }  
38    button{  
39    outline:0;  
40    border:0;  
41    background-color: darkred;  
42    color: black;  
43    width:100px;  
44    height:40px;  
45    }  
46    button:hover{  
47    background-color: brown;  
48    border:solid 1px black;
```

```
49    }  
50    select {  
51    font-size:1.3em;  
52    width:80%;  
53    text-align:center;  
54    | text-indent: 31%;  
55    | border: 2px solid black;  
56    |  
57    | color: #808000;  
58    |  
59    }  
60    hr{  
61    border-top: 1px dashed black;  
62    }  
63    </style>  
64    <head>  
65    <title>-- CO2 Cars Emission Predictor -- </title>  
66    </head>  
67    <body>  
68    <div class='idiv'>  
69    <br/>  
71    <h1>CO2 Cars Emission Predictor</h1>  
72    <hr/>
```

```
73    <br/>
74    <form action="/predict" method="POST">
75      <select id="Make" name="Make" >
76        <option>Select the type of CAR</option>
77        <option value="9">FORD</option>
78        <option value="5">CHEVROLET</option>
79        <option value="8">DODGE</option>
80        <option value="2">BMW</option>
81        <option value="10">GMC</option>
82        <option value="31">TOYOTA</option>
83        <option value="32">VOLKSWAGEN</option>
84        <option value="21">MERCEDES-BENZ</option>
85        <option value="6">CHRYSLER</option>
86        <option value="1">AUDI</option>
87        <option value="20">MAZDA</option>
88        <option value="33">VOLVO</option>
89        <option value="22">NISSAN</option>
90        <option value="25">PONTIAC</option>
91        <option value="27">SAAB</option>
92        <option value="28">SATURN</option>
93        <option value="30">SUZUKI</option>
94        <option value="11">HONDA</option>
95        <option value="29">SUBARU</option>
96        <option value="12">HYUNDAI</option>
```

```
97        <option value="0">ACURA</option>
98        <option value="7">DAEWOO</option>
99        <option value="15">JAGUAR</option>
100       <option value="17">KIA</option>
101     </select><br>
102     <select id="Vehicle_Class" name="Vehicle_Class">
103       <option value="">Select the Car Class</option>
104       <option value="0">COMPACT</option>
105       <option value="6">PICKUP TRUCK - STANDARD</option>
106       <option value="10">SUV</option>
107       <option value="2">MID-SIZE</option>
108       <option value="9">SUBCOMPACT</option>
109       <option value="11">TWO-SEATER</option>
110       <option value="7">STATION WAGON - MID-SIZE</option>
111       <option value="1">FULL-SIZE</option>
112       <option value="12">VAN - CARGO</option>
113       <option value="13">VAN - PASSENGER</option>
114       <option value="3">MINICOMPACT</option>
115       <option value="4">MINIVAN</option>
116       <option value="8">STATION WAGON - SMALL</option>
117       <option value="5">PICKUP TRUCK - SMALL</option>
118     </select><br>
119     <input class="form-input" type="text" name="Engine_Size" placeholder="Enter the engine size"><br>
120     <input class="form-input" type="text" name="Cylinders" placeholder="Enter the Cylinders size"><br>
```

```

121  <select id="Transmission" name="Transmission">
122    <option value="">Select transmission of the car</option>
123      <option value="0">Automatic</option>
124      <option value="3">Manual</option>
125    <option value="1">Automatic with Select Shift</option>
126      <option value="2">Continuously Variable</option>
127    </select><br>
128  <select id="Fuel_Type" name="Fuel_Type">
129    <option value="">Select the Fuel Type</option>
130      <option value="4">Regular Gasoline</option>
131      <option value="3">Premium Gasoline</option>
132      <option value="0">Diesel</option>
133      <option value="1">Ethanol(E85)</option>
134      <option value="2">Natural Gas</option>
135    </select><br>
136  <input class="form-input" type="text" name="Fuel_Consumption_City">
137  placeholder="Enter the Fuel Consumption City(L/100km)"<br>
138  <input class="form-input" type="text" name="Fuel_Consumption_Hwy">
139  placeholder="Enter the Fuel Consumption Highway(L/100km)"<br>
140  <input class="form-input" type="text" name="Fuel_Consumption_Comb(mpg)">
141  placeholder="Enter the Combine Fuel Consumption(mpg)"<br>
142  <button type="submit" class="btn btn-primary btn-block btn-large">Predict</button>
143  </form>
144  <br/>
145  <br/>
146  <br/>
147  </div>
148
149  </body>
150  </html>

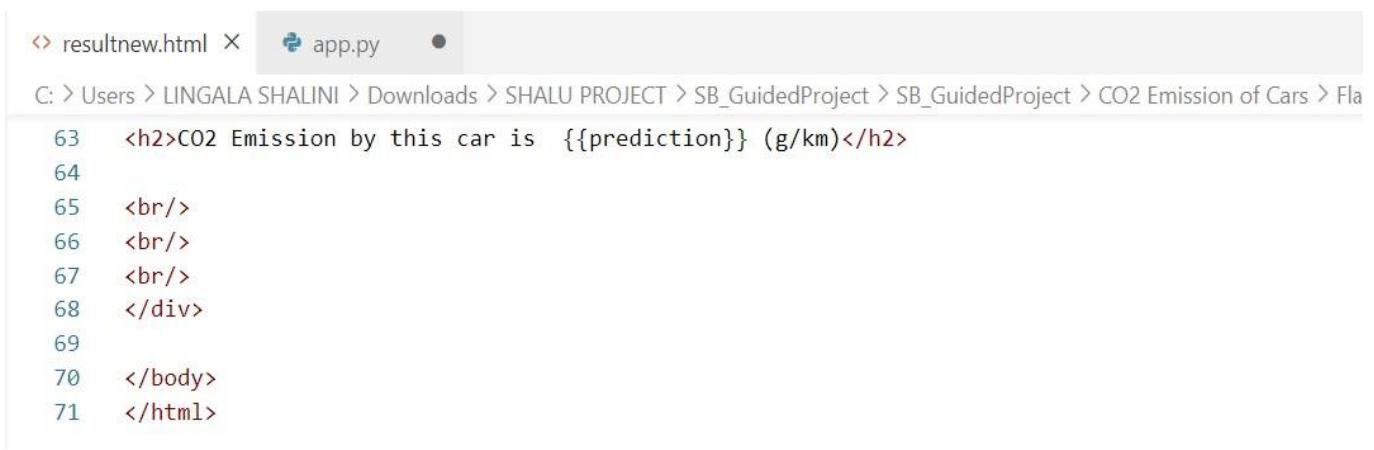
```

Figure 19: result.html is the page which displays the result that emission CO₂ releases from the vehicles.

3.result.html

```
<> resultnew.html < app.py ●
C: > Users > LINGALA SHALINI > Downloads > SHALU PROJECT > SB_GuidedProject > SB_GuidedProject > CO2 Emission of Cars > Flask > templates > <> resultnew.html >
1  <html>
2  <style>
3  .idiv{
4    width:60%;
5    margin:auto;
6    text-align:center;
7    margin-top:2%;
8    border-radius:10px;
9    background-repeat: no-repeat;
10
11
12
13  margin-top:2%;
14 }
15 body{
16  background-color: black;
17  font-family:segoe ui;
18  background: linear-gradient(rgba(0,0,0,0.8),rgba(0,0,0,0.8)),url(https://cdn.pixabay.com/photo/2015/10/01/13/36/car-914411\_960\_720.jpg);
19  height: 100vh;
20  -webkit-background-size: cover;
21  background-size: cover;
22  background-position: center center;
23  position: relative;
24 }
25 input{
26  font-size:1.3em;
27  width:80%;
28  text-align:center;
29 }
30 input placeholder{
31 }
```

```
<> resultnew.html < app.py ●
C: > Users > LINGALA SHALINI > Downloads > SHALU PROJECT > SB_GuidedProject > SB_GuidedProject > CO2 Emission of Cars > Flask > tem
32  text-align:center;
33 }
34 button{
35  outline:0;
36  border:0;
37  background-color: darkred;
38  color: white;
39  width:100px;
40  height:40px;
41 }
42 button:hover{
43  background-color: brown;
44  border:solid 1px black;
45 }
46 h1{
47  color: red;
48 }
49 h2{
50  color: olive;
51 }
52 }
53 </style>
54 <head>
55 <title>-- CO2 Cars Emission Predictor -- </title>
56 </head>
57 <body>
58 <div class='idiv'>
59
60 <br/>
61 <h1>CO2 Cars Emission Predictor</h1>
62 <br/>
```

The screenshot shows a code editor interface with two tabs: 'resultnew.html' and 'app.py'. The 'resultnew.html' tab is active, displaying the following HTML code:

```
C: > Users > LINGALA SHALINI > Downloads > SHALU PROJECT > SB_GuidedProject > SB_GuidedProject > CO2 Emission of Cars > Fla
63   <h2>CO2 Emission by this car is {{prediction}} (g/km)</h2>
64
65   <br/>
66   <br/>
67   <br/>
68   </div>
69
70   </body>
71   </html>
```

Figure 20: result.html is the page which is code for app.py.

4. CONCLUSION

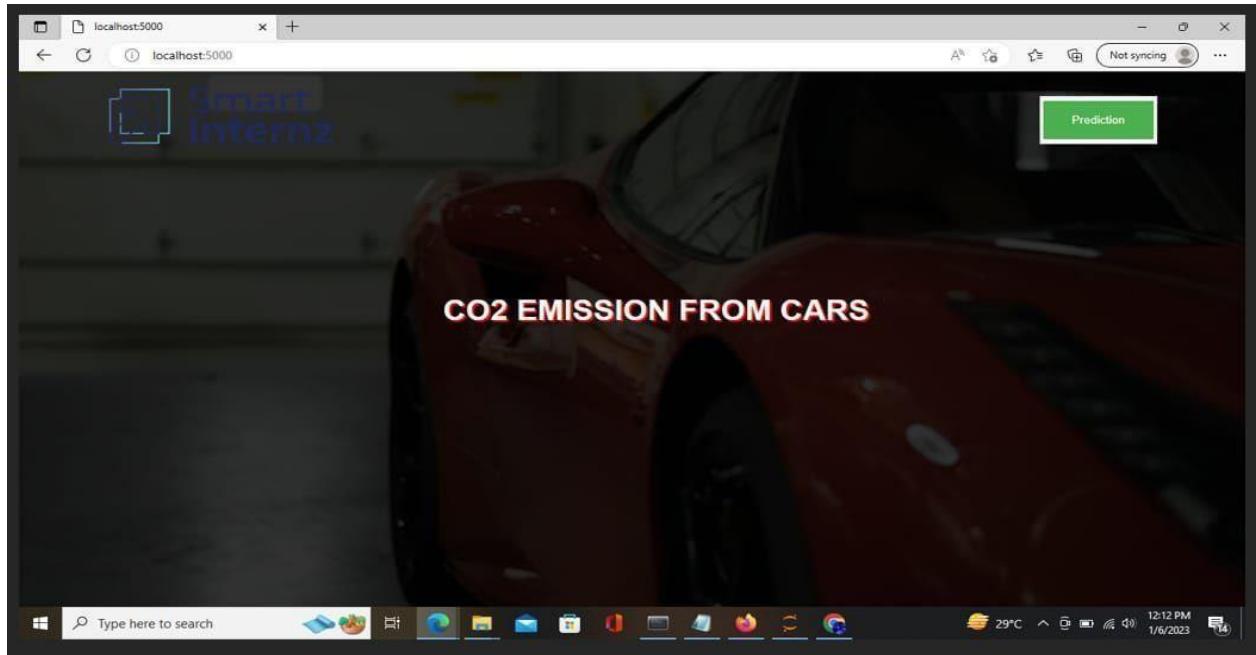


Figure 21: Home page

A screenshot of the CO2 Cars Emission Predictor input page. The title 'CO2 Cars Emission Predictor' is at the top. Below it is a form with the following fields:

- Select the type of CAR
- Select the Car Class
- Enter the engine size
- Enter the Cylinders size
- Select transmission of the car
- Select the Fuel TypeRegular Gasoline
- Enter the Fuel Consumption City(L/100km)
- Enter the Fuel Consumption Highway(L/100km)
- Enter the Combine Fuel Consumption(mpg)

A red 'Predict' button is located at the bottom of the form.

Figure 22: Input page

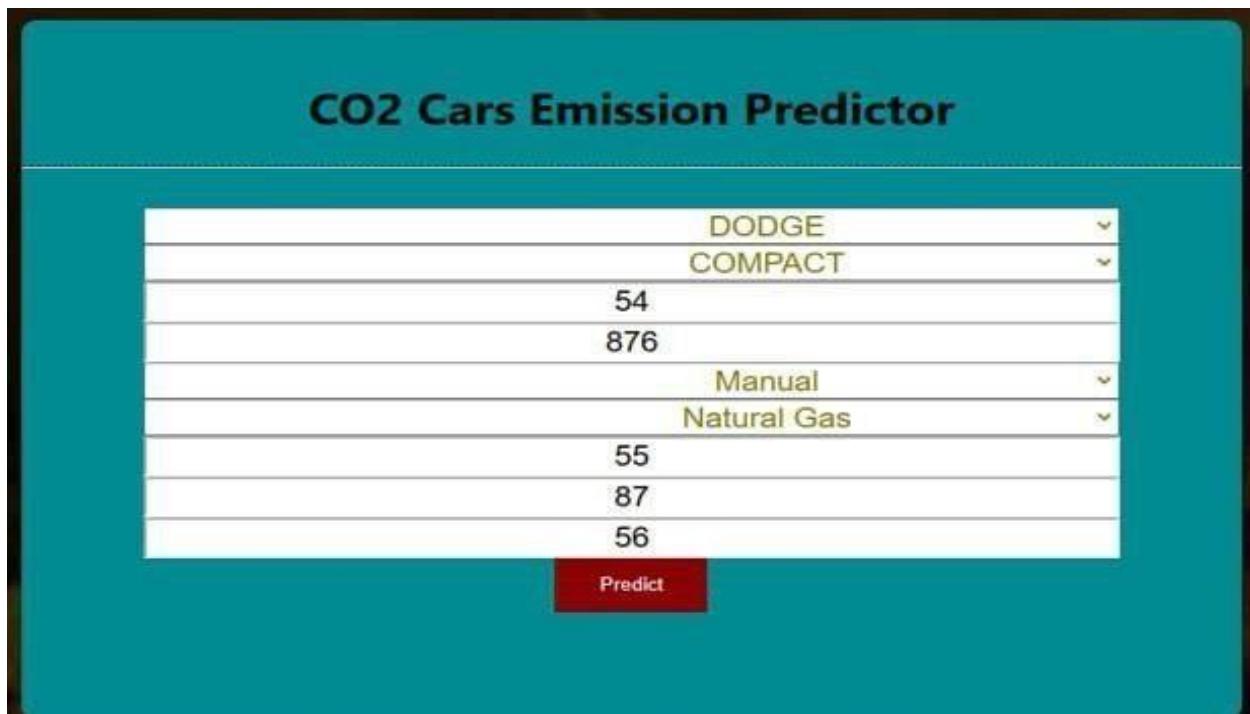


Figure 23: prediction

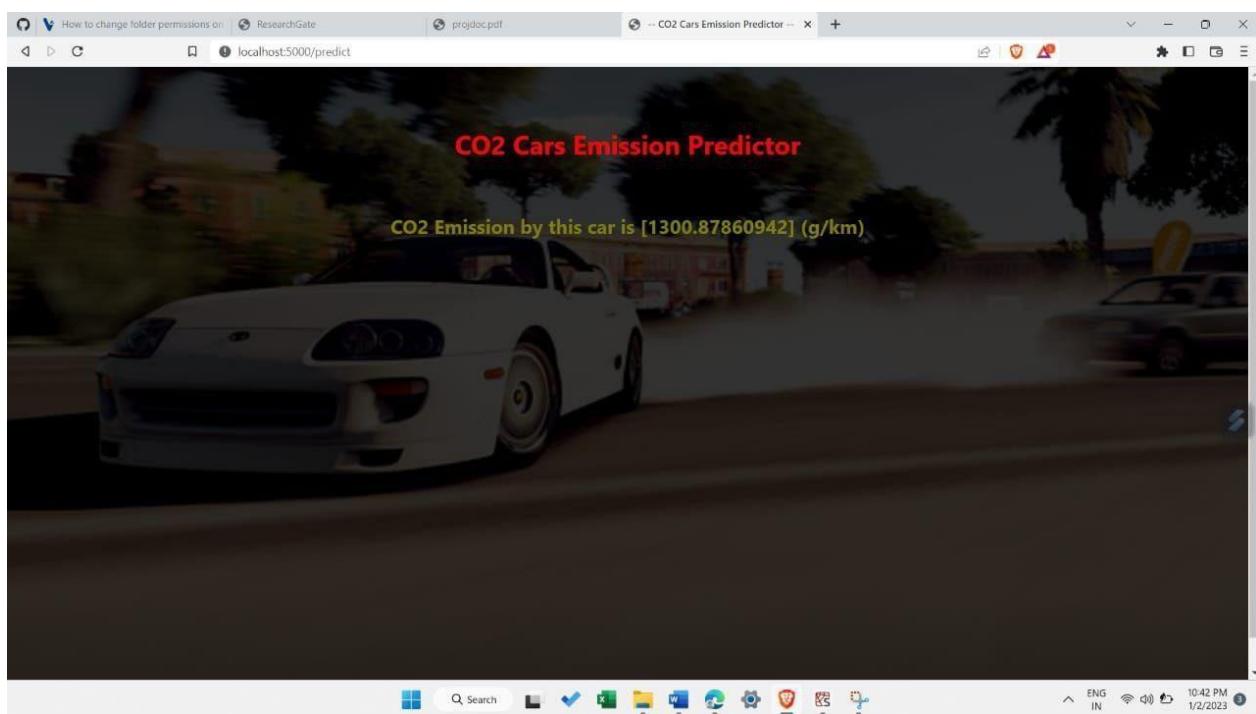


Figure 24: output page

5. ADVANTAGES

1. Technology more mature than other alternatives; can easily retrofit into existing plants; High CO₂ concentration enhance sorption efficiency; fully developed technology, commercially deployed at the required scale in some industrial sectors; opportunity for retrofit to existing plants.
2. Very high CO₂ concentration that enhances absorption efficiency; mature air separation technologies available; reduced volume of gas to be treated, hence required smaller boiler and other equipment; CO₂ is the main combustion product, which remains unmixed with N₂, thus avoiding energy intensive air separation.

6. DISADVANTAGES

1. Low CO₂ concentration affects the capture efficiency.
2. Temperature associated heat transfer problem and efficiency decay issues associated with the use of hydrogen-rich gas turbine fuel; high parasitic power requirement for sorbent regeneration; High efficiency drop and energy penalty; cryogenic O₂ production is costly; corrosion problem may arise.

7. BIBILOGRAPHY

1. U.S. EPA (March 2009), “Greenhouse Gas Emissions from the U.S. Transportation Sector,1990-2003” (found at: <http://www.epa.gov/otaq/climate/420r06003.pdf>).
2. Barkenbus, J. N. “Eco-driving: an overlooked climate change initiative”, Energy Policy, 2010,38 (2010) 762-769.
3. Alessandrini, A.; Orecchini, F.; Ortenzi, F.; Villatico Campbell F., “Drive-style emissionstesting on the latest two Honda hybrid technologies”, European Conference of Transport Research Institutes (ECTRI), 2009, DOI 10.1007/s12544-009-0008-3.
4. Ecodrive.org, 2009 (found at: <http://www.ecodrive.org>).
5. Alliance of Automobile Manufacturers, Ecodriving USA, 2008 (found at:<http://www.ecodrivingusa.com>).
6. Alliance to Save Energy, The Drive Smarter Challenge, 2008 (found at:<http://drivesmarterchallenge.org>).
7. “Denver’s driving change program reduces vehicular CO2 emissions”, Enviance, 2009, January 27 (found at:
<http://www.enviance.com/about-enviance/PressReleaseView.aspx?id=53>).
8. International Transport Forum, Ecodriving, 2007 (found at: <http://www.internationaltransportforum.org/Proceedings/ecodriving/ecodriving07.html>).
9. CIECA, “Internal project on “Eco-driving” in category B driver training & the driving test, Final Report”, 2007, November 5 (found at: <http://www.thepep.org/ClearingHouse/docfiles/CIECA.internal.project.on.Eco-driving.pdf>).
10. Vermeulen, R. J. “The effects of a range of measures to reduce the tail pipe emissions and/orthe fuel consumption of modern passenger cars on petrol and diesel”, TNO Report, 2006, December 22.

8. HELP FILE

PROJECT EXECUTION:

STEP-1: Go to Start, Open GOOGLE CHROME.

STEP-2: After Opening GOOGLE CHROME, Then Search for GOOGLE COLAB.

STEP-3: Open “Major project code” IPYNB file.

STEP-4: Then run all the cells.

STEP-5: All the data preprocessing, training and testing, model building, accuracy of the model can be showcased.

STEP-6: And a pickle file will be generated.

STEP-7: Create a Folder named FLASK . Extract the pickle file into this Flask Folder.

STEP-8: Extract all the html files (home.html, index1.html, indexnew.html, result.html) and python file(app.py) into the FLASK Folder.

STEP-9: Then Open COMMAND PROMPT.

STEP-10: After Opening follow the below steps: cd File Path◊click enter pythonapp.py◊click enter (We could see running of files).

STEP-11: Then open BROWSER, at the URL area type —http://127.0.0.1:5000/”.

STEP-12: Home page of the project will be displayed.

STEP-13: Click on —Go to Predict”. Directly it will be navigated to index page.

STEP-14: An index page will be displayed where the user needs to give the inputs and then click on —Predict. Output will be generated what amount of fuel consumption is taking place.